INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN FLUIDS
Int. J. Numer. Meth. Fluids 2004; **44**:483–503 (DOI: 10.1002/fld.648)

A stabilized finite element predictor–corrector scheme for the incompressible Navier–Stokes equations using a nodal-based implementation

R. Codina*,† and A. Folch‡

Universitat Politècnica de Catalunya, Jordi Girona 1-3, Edifici C1, 08034 Barcelona, Spain

SUMMARY

A finite element model to solve the incompressible Navier–Stokes equations based on the stabilization with orthogonal subscales, a predictor–corrector scheme to segregate the pressure and a nodal based implementation is presented in this paper. The stabilization consists of adding a least-squares form of the component orthogonal to the finite element space of the convective and pressure gradient terms, which allows to deal with convection-dominated flows and to use equal velocity–pressure interpolation. The pressure segregation is inspired in fractional step schemes, although the converged solution corresponds to that of a monolithic time integration. Finally, the nodal-based implementation is based on an *a priori* calculation of the integrals appearing in the formulation and then the construction of the matrix and right-hand side vector of the final algebraic system to be solved. After appropriate approximations, this matrix and this vector can be constructed directly for each nodal point, without the need to loop over the elements and thus making the calculations much faster. Some issues related to this implementation for fractional step and our predictor–corrector scheme, which is the main contribution of this paper, are discussed. Copyright © 2004 John Wiley & Sons, Ltd.

KEY WORDS: stabilized FEM; fractional step schemes; predictor–corrector methods; nodal based implementation

1. INTRODUCTION

It is well known that the pressure offers two major problems in the numerical approximation of the incompressible Navier–Stokes equations. First, if the standard Galerkin method is used, its interpolation must be different from that of the velocity in order to satisfy the classical inf–sup condition. On the other hand, the velocity–pressure coupling makes the solution of the linear system arising after the discretization very costly.

Although there are several velocity–pressure pairs that satisfy the inf–sup condition, it is also possible to use equal interpolation provided the standard Galerkin method is modified

*Correspondence to: R. Codina, Universitat Politècnica de Catalunya, Jordi Girona 1-3, Edifici C1, 08034 Barcelona, Spain.
†E-mail: ramon.codina@upc.es
‡E-mail: folch@cimne.upc.es

Received 29 March 2003
Revised 22 July 2003

by a *stabilization method*. Examples of these are the methods in References [1, 2], the Galerkin/least-squares (GLS) technique [3–5] and least-squares methods for first-order systems as those in Reference [6] among others.

Regarding the velocity–pressure coupling, fractional step methods for the incompressible Navier–Stokes equations have enjoyed widespread popularity since the original works of Chorin [7] and Temam [8]. The reason for this relies on the computational efficiency of these methods (see e.g. References [9–11]), basically due to the uncoupling of the pressure from the velocity components.

In this paper, we describe a finite element method able to deal with *equal* velocity pressure interpolations. On the other hand, we present an iterative algorithm that allows to uncouple the calculation of the pressure from that of the velocity, motivated by what is commonly done in fractional step methods, although the converged solution of the iterative procedure is that of the *monolithic* (coupled velocity–pressure) time discretization. In this sense, our approach can be viewed as a *predictor–(multi)corrector* method, in the spirit of Reference [12] (see also references therein). A very similar method was presented in Reference [13], the only difference being the treatment of the stabilizing term described next.

The stabilized method we use is not only intended to allow equal velocity–pressure interpolation, but also to deal with highly convective flows. It is based on the subgrid scale concept and, in particular, in the approach introduced by Hughes in References [14, 15] for the scalar convection–diffusion equation. The basic idea is to approximate the *effect* of the component of the continuous solution which cannot be resolved by the finite element mesh on the discrete finite element solution. An important feature of the formulation developed herein is that the unresolved component, hereafter referred to as *subgrid scale or subscale*, is assumed to be L^2-orthogonal to the finite element space, in a sense which is explained later. A slightly modified version of the method to be presented here (with very similar properties) can be found in Reference [13].

The previous two points (stabilization and predictor–corrector scheme) can be considered as *formulation* issues. However, we also discuss a non-standard *implementation*, which was originally proposed in Reference [16] and that we apply here to a different finite element formulation, with its own peculiarities. After computing the volume integrals of the products of the shape functions and its derivatives, the system matrix and force vector of the resulting algebraic system are obtained from them. This is done at each iteration and at each time step, without the need to recompute volume integrals any more. However, some approximations are required to do this. These approximations and their implications are described in this paper.

Whereas for a standard finite element implementation the normal flow of the calculation involves a loop over the elements, the calculation of the element contributions (to the system matrix and to the force vector) and the assembly of these into the global arrays, the flow of the calculations for the algorithm presented here is very different. The system matrix is formed of block matrices corresponding to the nodal points that can be obtained directly, without any reference to the elements. This is done by looping over the nodes of the finite element mesh and then over the nodes connected to a given nodal point. This involves the storage of the graph of the mesh, as well as the graph of the boundary mesh when the contributions from the boundaries need to be accounted for.

We have organized the paper as follows. In the following section we present the problem to approximate, some notation and the time discretization we will use. In Section 3, we summarize the stabilized finite element formulation and in Section 4, we describe the monolithic

and the predictor–corrector strategy we propose, together with some computational aspects of the algorithm. In Section 5, we summarize the nodal based implementation proposed in Reference [16] and discuss some particular details of its application to the formulation proposed herein. Some numerical results are presented in Section 6 and finally some conclusions are drawn in Section 7.

2. PROBLEM STATEMENT

2.1. Continuous problem

Let Ω be the domain in $\mathbb{R}^{n_{sd}}$ occupied by the fluid, where $n_{sd}=2$ or 3 is the number of space dimensions, $\Gamma=\partial\Omega$ its boundary and $[0,T]$ the time interval of analysis. The Navier–Stokes problem consists of finding a velocity $\mathbf{u}$ and a pressure p such that

$$\partial_t\mathbf{u}+\mathbf{u}\cdot\nabla\mathbf{u}-\nu\nabla^2\mathbf{u}+\nabla p=\mathbf{f} \quad \text{in } \Omega,\ t\in(0,T) \tag{1}$$

$$\nabla\cdot\mathbf{u}=0 \quad \text{in } \Omega,\ t\in(0,T) \tag{2}$$

$$\mathbf{u}=\mathbf{0} \quad \text{on } \Gamma,\ t\in(0,T) \tag{3}$$

$$\mathbf{u}=\mathbf{u}^0 \quad \text{in } \Omega,\ t=0 \tag{4}$$

where ν is the kinematic viscosity, $\mathbf{f}$ is the force vector and $\mathbf{u}^0$ is the velocity initial condition. We have considered the homogeneous Dirichlet boundary condition (3) for simplicity.

To write the weak form of problem (1)–(4) we need to introduce some notation. We denote by $H^1(\Omega)$ the Sobolev space of functions whose first derivatives belong to $L^2(\Omega)$, and by $H^1_0(\Omega)$ the subspace of $H^1(\Omega)$ of functions with zero trace on Γ. A bold character is used for the vector counterpart of these spaces. The L^2 scalar product in a set ω is denoted by $(\cdot,\cdot)_\omega$, and the L^2 norm by $\|\cdot\|_\omega$. The subscript ω is omitted when it coincides with Ω. To pose the problem we also need the functional spaces $\mathbf{V}_{st}=\mathbf{H}^1_0(\Omega)^{n_{sd}}$ and $Q_{st}=\{q\in L^2(\Omega)\mid\int_\Omega q=0\}$, as well as $\mathbf{V}=\mathbf{L}^2(0,T;\mathbf{V}_{st})$ and $Q=L^2(0,T;Q_{st})$ for the transient problem.

Assuming for simplicity the force vector to be square integrable, the weak form of problem (1)–(4) consists of finding $(\mathbf{u},p)\in\mathbf{V}\times Q$ such that

$$(\partial_t\mathbf{u}+\mathbf{u}\cdot\nabla\mathbf{u},\mathbf{v})+\nu(\nabla\mathbf{u},\nabla\mathbf{v})-(p,\nabla\cdot\mathbf{v})=(\mathbf{f},\mathbf{v}) \tag{5}$$

$$(q,\nabla\cdot\mathbf{u})=0 \tag{6}$$

for all $\langle\mathbf{v},q\rangle\in\mathbf{V}_{st}\times Q_{st}$, and satisfying the initial condition in a weak sense.

2.2. Time discretization

Any time integration of (5)–(6) is, in principle possible. However, we shall concentrate on the monolithic trapezoidal rule (solving for the velocity and the pressure at the same time). The time discretized version of (5) and (6) in this case consists of solving the following problem: from known $\mathbf{u}^n$, find $\mathbf{u}^{n+1}\in\mathbf{V}_{st}$ and $p^{n+1}\in Q_{st}$ such that

$$(\delta^n_t\mathbf{u}+\mathbf{u}^{n+\theta}\cdot\nabla\mathbf{u}^{n+\theta},\mathbf{v})+\nu(\nabla\mathbf{u}^{n+\theta},\nabla\mathbf{v})-(p^{n+\theta},\nabla\cdot\mathbf{v})=(\bar{\mathbf{f}}^{n+\theta},\mathbf{v}) \tag{7}$$

$$(q,\nabla\cdot\mathbf{u}^{n+\theta})=0 \tag{8}$$

for all $(\mathbf{v},q)\in\mathbf{V}_{\rm st}\times Q_{\rm st}$, where δt is the time step size, superscript m refers to the time step level $t^m=m\,\delta t, \theta\in(0,1]$ and we use the notation

$$\mathbf{u}^{n+\theta}:=\theta\mathbf{u}^{n+1}+(1-\theta)\mathbf{u}^n,\quad \delta\mathbf{u}^n:=\mathbf{u}^{n+1}-\mathbf{u}^n \quad\text{and}\quad \delta_t^n\mathbf{u}:=\frac{\delta\mathbf{u}^n}{\delta t}$$

The force term $\bar{\mathbf{f}}^{n+\theta}$ in (7) and below has to be understood as the time average of the force in the interval $[t^n,t^{n+1}]$, even though we use a superscript $n+\theta$ to characterize it. The pressure value computed here has been identified as the pressure evaluated at $t^{n+\theta}$, although this is irrelevant for the velocity approximation. The values of interest of θ are $\theta=\frac{1}{2}$, corresponding to the second-order Crank–Nicolson scheme, and $\theta=1$, which corresponds to the backward Euler method and which is the choice adopted in the two examples of this paper.

3. STABILIZATION WITH ORTHOGONAL SUBSCALES

In this section, we present the basic finite element formulation, which was already proposed in References [13, 17]. Let $\mathcal{T}_h$ denote a finite element partition of the domain Ω of diameter h, from which we construct the finite element spaces $Q_h, \mathbf{V}_h$ and $\mathbf{V}_{h0}$, approximations to $Q_{\rm st}$, $\mathbf{H}^1(\Omega)^{n_{\rm sd}}$ and $\mathbf{V}_{\rm st}$, respectively. The former is made up with continuous functions of degree $k_{\rm q}$ and the other two with continuous vector functions of degree $k_{\rm v}$, the latter verifying the homogeneous Dirichlet boundary conditions. In the following, finite element functions will be identified with a subscript h.

The discrete problem is obtained by approximating $\mathbf{u}$ and p. If $\mathbf{u}_h$ and p_h are the finite element unknowns, we approximate $\mathbf{u}\approx\mathbf{u}_h+\tilde{\mathbf{u}}$ and $p\approx p_h$, that is, the velocity is approximated by its finite element component plus an additional term that we call *subgrid scale* or *subscale* (for lack of a better name), and the pressure will be taken as the finite element one. We shall $\mathbf{u}^n\approx\mathbf{u}_*^n:=\mathbf{u}_h^n+\tilde{\mathbf{u}}^n$ and $p^n\approx p_h^n$ the velocity and the pressure for time level n. Considering the spatial interpolation, we assume that $\mathbf{u}_h^n$ and p_h^n are constructed using the standard finite element interpolation. In particular, equal velocity–pressure interpolation is possible with the formulation to be presented.

The important point is the behaviour assumed for $\tilde{\mathbf{u}}^n$. To simplify the discussion, we assume that it vanishes on the interelement boundaries, that is, it is a bubble-like function [18, 19]. However, contrary to what is commonly done, we do not assume any particular behaviour of $\tilde{\mathbf{u}}^n$ within the element domains. We will indicate later on how to approximate it.

If in (7) $\mathbf{u}^n$ is replaced by $\mathbf{u}_*^n:=\mathbf{u}_h^n+\tilde{\mathbf{u}}^n$, p^n is replaced by p_h^n, the terms involving $\tilde{\mathbf{u}}^n$ are integrated by parts, and the test functions are taken in the finite element space, one gets

$$(\delta_t^n\mathbf{u}_h+\mathbf{u}_*^{n+\theta}\cdot\nabla\mathbf{u}_h^{n+\theta},\mathbf{v}_h)+\nu(\nabla\mathbf{u}_h^{n+\theta},\nabla\mathbf{v}_h)-(p_h^{n+\theta},\nabla\cdot\mathbf{v}_h)$$
$$+(\delta_t^n\tilde{\mathbf{u}},\mathbf{v}_h)-(\tilde{\mathbf{u}}^{n+\theta},\nu\nabla_h^2\mathbf{v}_h+\mathbf{u}_*^{n+\theta}\cdot\nabla\mathbf{v}_h)=(\bar{\mathbf{f}}^{n+\theta},\mathbf{v}_h) \tag{9}$$

$$(q_h,\nabla\cdot\mathbf{u}_h^{n+\theta})-(\tilde{\mathbf{u}}^{n+\theta},\nabla q_h)=0 \tag{10}$$

which must hold for all $\mathbf{v}_h \in \mathbf{V}_{h,0}$ and $q_h \in Q_h$. The notation ∇_h^2 is used to indicate that the Laplacian needs to be evaluated element by element. It is important to note that the advection velocity in (9) is $\mathbf{u}_*^{n+\theta}$.

The equation for the subscales $\tilde{\mathbf{u}}^{n+1}$ is obtained by taking the velocity test function in (7) in its space. The result is that, within each element [20]:

$$\delta_t \tilde{\mathbf{u}}^n + \mathbf{u}_*^{n+\theta} \cdot \nabla \tilde{\mathbf{u}}^{n+\theta} - \nu \nabla^2 \tilde{\mathbf{u}}^{n+\theta} = \mathbf{r}^{n+\theta} + \mathbf{v}_{h,\mathrm{ort}}^{n+\theta} \tag{11}$$

$$\mathbf{r}^{n+\theta} := \mathbf{f}^{n+\theta} - (-\nu \nabla^2 \mathbf{u}_h^{n+\theta} + \mathbf{u}_*^{n+\theta} \cdot \nabla \mathbf{u}_h^{n+\theta} + \nabla p_h^{n+\theta}) \tag{12}$$

where $\mathbf{v}_{h,\mathrm{ort}}^{n+\theta}$ is a function L^2-orthogonal to the space of subscales.

The next step is *to model* the solution $\tilde{\mathbf{u}}^{n+1}$ of (11). This means to give a closed-form expression for it that approximates the exact solution in some sense. It is shown in Reference [17] that if we replace (11) by the algebraic equation

$$\left(\frac{1}{\theta \delta t} + \frac{1}{\tau}\right) \tilde{\mathbf{u}}^{n+\theta} = \frac{1}{\theta \delta t} \tilde{\mathbf{u}}^n + \mathbf{r}^{n+\theta} + \mathbf{v}_{h,\mathrm{ort}}^{n+\theta} \tag{13}$$

with

$$\tau := \left(c_1 \frac{\nu}{h^2} + c_2 \frac{|\mathbf{u}_*^{n+\theta}|}{h}\right)^{-1} \tag{14}$$

then there are values of the constants c_1 and c_2 (which do not depend neither on the discretization nor on the equation coefficients ν and $|\mathbf{u}_*^{n+\theta}|$) for which the solutions of (11) and (13) have approximately the same L^2-norm over the elements. Note that in (13) we do not require the subscales to vanish on the element boundaries. In the numerical calculations, we always use $c_1 = 4$ and $c_2 = 2$.

It still remains to define the space of the subscales. A main feature of our formulation is that we take it (approximately) orthogonal to the finite element space. Imposing this in (13) allows to compute $\mathbf{v}_{h,\mathrm{ort}}^{n+\theta}$, which turns out to be minus the projection of $\mathbf{r}^{n+\theta}$ onto the finite element space $\mathbf{V}_h$. Therefore,

$$\left(\frac{1}{\theta \delta t} + \frac{1}{\tau}\right) \tilde{\mathbf{u}}^{n+\theta} = \frac{1}{\theta \delta t} \tilde{\mathbf{u}}^n + P_h^{\perp}(\mathbf{r}^{n+\theta}) \tag{15}$$

where $P_h^{\perp} = I - P_h$, P_h being the L^2-projection onto $\mathbf{V}_h$.

To complete the description of the method, we will make two further approximations. First, we will consider that $P_h^{\perp}(\mathbf{f}) = \mathbf{0}$, that is, $\mathbf{f}$ is a finite element function. This does not alter the accuracy of the final formulation. Secondly, we will neglect the orthogonal projection of viscous term in $\mathbf{r}^{n+\theta}$ and $\nu \nabla_h^2 \mathbf{v}_h$ in (9). This is exact for linear elements and leads to a consistent formulation for higher order elements. Thus, the final system of equations we are left with is

$$\begin{aligned}
&(\delta_t^n \mathbf{u}_h + \mathbf{u}_*^{n+\theta} \cdot \nabla \mathbf{u}_h^{n+\theta}, \mathbf{v}_h) + \nu(\nabla \mathbf{u}_h^{n+\theta}, \nabla \mathbf{v}_h) - (p_h^{n+\theta}, \nabla \cdot \mathbf{v}_h) \\
&\quad + (\tau_t P_h^{\perp}(\mathbf{u}_*^{n+\theta} \cdot \nabla \mathbf{u}_h^{n+\theta} + \nabla p_h^{n+\theta}), \mathbf{u}_*^{n+\theta} \cdot \nabla \mathbf{v}_h) \\
&\quad = (\bar{\mathbf{f}}^{n+\theta}, \mathbf{v}_h) + \frac{1}{\theta \delta t}(\tau_t \tilde{\mathbf{u}}^n, \mathbf{u}_*^{n+\theta} \cdot \nabla \mathbf{v}_h)
\end{aligned} \tag{16}$$

$$(q_h, \nabla \cdot \mathbf{u}_h^{n+\theta}) + (\tau_t P_h^{\perp}(\mathbf{u}_*^{n+\theta} \cdot \nabla \mathbf{u}_h^{n+\theta} + \nabla p_h^{n+\theta}), \nabla q_h)$$
$$= \frac{1}{\theta \delta t}(\tau_t \tilde{\mathbf{u}}^n, \nabla q_h) \tag{17}$$

where

$$\tau_t := \left(\frac{1}{\theta \delta t} + \frac{1}{\tau} \right)^{-1}$$

Note that the term $(\delta_t^n \tilde{\mathbf{u}}, \mathbf{v}_h)$ in (9) vanishes because of the orthogonality of $\delta_t^n \tilde{\mathbf{u}}$ and $\mathbf{v}_h$. Note also that the parameter τ_t has been included within the inner product, since in principle it changes from point to point. The terms multiplied by this parameter must be responsible for the enhancement of stability with respect to the standard Galerkin method; we will call them *stabilization terms*.

Once arrived to the final problem (16)–(17), we can make some further modifications *provided the consistency of the method is maintained*, that is to say, these modifications do not alter the fact that the exact solution is still a solution of the discrete problem. For the discussion of the next section and the numerical results, we will make two more modifications:

- We will consider the subscales *quasi-static*. As explained in Reference [17], this leads to $\tilde{\mathbf{u}}^{n+\theta} = \tau P_h^{\perp}(\mathbf{r}^{n+\theta})$ as the solution of (15).
- The advection velocity $\mathbf{u}_*$ will be replaced $\mathbf{u}_h$. This means that we neglect the influence of the subscales in the transport of momentum. However, this point needs to be further explored, since this influence might be the key of turbulence modelling.

With these modifications, we arrive at the discrete problem

$$(\delta_t^n \mathbf{u}_h + \mathbf{u}_h^{n+\theta} \cdot \nabla \mathbf{u}_h^{n+\theta}, \mathbf{v}_h) + \nu(\nabla \mathbf{u}_h^{n+\theta}, \nabla \mathbf{v}_h) - (p_h^{n+\theta}, \nabla \cdot \mathbf{v}_h)$$
$$+(\tau P_h^{\perp}(\mathbf{u}_h^{n+\theta} \cdot \nabla \mathbf{u}_h^{n+\theta} + \nabla p_h^{n+\theta}), \mathbf{u}_h^{n+\theta} \cdot \nabla \mathbf{v}_h) = (\bar{\mathbf{f}}^{n+\theta}, \mathbf{v}_h), \quad \forall \mathbf{v}_h \in \mathbf{V}_{h,0} \tag{18}$$

$$(q_h, \nabla \cdot \mathbf{u}_h^{n+\theta}) + (\tau P_h^{\perp}(\mathbf{u}_h^{n+\theta} \cdot \nabla \mathbf{u}_h^{n+\theta} + \nabla p_h^{n+\theta}), \nabla q_h) = 0, \quad \forall q_h \in Q_h \tag{19}$$

In the following, we will describe the matrix form of the problem and the predictor–corrector scheme, whereas in Section 5 we will present a nodal based implementation to compute all the terms appearing in (19).

4. MONOLITHIC AND PREDICTOR–CORRECTOR SCHEMES

We start this section by presenting the matrix version of the monolithic problem. After this, we will present a fractional step method that will lead us to the *predictor–multicorrector* method we propose. Again, this section follows with minor modifications the method proposed in Reference [13], the difference being the treatment of the stabilizing terms.

4.1. Matrix version of the problem

Let us note first that the orthogonal projection of the convective and pressure gradient terms can be written as

$$P_h^{\perp}(\mathbf{u}_h^{n+\theta}\cdot\nabla\mathbf{u}_h^{n+\theta}+\nabla p_h^{n+\theta})=\mathbf{u}_h^{n+\theta}\cdot\nabla\mathbf{u}_h^{n+\theta}+\nabla p_h^{n+\theta}-\mathbf{y}_h^{n+\theta}$$

where $\mathbf{y}_h^{n+\theta}$ is the solution of

$$(\mathbf{y}_h^{n+\theta},\mathbf{v}_h)=(\mathbf{u}_h^{n+\theta}\cdot\nabla\mathbf{u}_h^{n+\theta}+\nabla p_h^{n+\theta},\mathbf{v}_h),\quad\forall\mathbf{v}_h\in\mathbf{V}_h \tag{20}$$

From this expression it is easily checked that the discrete variational problem (18) and (19), together with the projection equation (20), lead to the non-linear algebraic system

$$\mathsf{M}\delta_t^n\mathsf{U}+\mathsf{K}(\mathsf{U}^{n+\theta})\mathsf{U}^{n+\theta}+\mathsf{G}\mathsf{P}^{n+\theta}+\mathsf{S}_{uu}\mathsf{U}^{n+\theta}+\mathsf{S}_{up}\mathsf{P}^{n+\theta}-\mathsf{S}_{uy}\mathsf{Y}^{n+\theta}=\mathsf{F}^{n+\theta} \tag{21}$$

$$\mathsf{D}\mathsf{U}^{n+\theta}+\mathsf{S}_{pp}\mathsf{P}^{n+\theta}+\mathsf{S}_{pu}\mathsf{U}^{n+\theta}-\mathsf{S}_{py}\mathsf{Y}^{n+\theta}=0 \tag{22}$$

$$\mathsf{M}\mathsf{Y}^{n+\theta}-\mathsf{C}(\mathsf{U}^{n+\theta})\mathsf{U}^{n+\theta}-\mathsf{G}\mathsf{P}^{n+\theta}=0 \tag{23}$$

where U, P and Y are the arrays of nodal unknowns for $\mathbf{u}$, p and $\mathbf{y}$, respectively. If we denote the node indexes with superscripts a,b, the space indexes with subscripts i,j, and the standard shape function of node a by N^a, the components of the arrays involved in these equations are:

$$\mathsf{M}_{ij}^{ab}=(N^a,N^b)\delta_{ij}\quad(\delta_{ij}\text{ is the Kronecker }\delta)$$

$$\mathsf{K}(\mathsf{U}^{n+\theta})_{ij}^{ab}=(N^a,\mathbf{u}_h^{n+\theta}\cdot\nabla N^b)\delta_{ij}+\nu(\nabla N^a,\nabla N^b)\delta_{ij}$$

$$\mathsf{G}_i^{ab}=(N^a,\partial_i N^b)$$

$$\mathsf{S}_{uu_{ij}}^{ab}=(\tau\mathbf{u}_h^{n+\theta}\cdot\nabla N^a,\mathbf{u}_h^{n+\theta}\cdot\nabla N^b)\delta_{ij}$$

$$\mathsf{S}_{up_i}^{ab}=(\tau\mathbf{u}_h^{n+\theta}\cdot\nabla N^a,\partial_i N^b)$$

$$\mathsf{S}_{uy_{ij}}^{ab}=(\tau\mathbf{u}_n^{n+\theta}\cdot\nabla N^a,N^b)\delta_{ij}$$

$$\mathsf{D}_j^{ab}=(N^a,\partial_j N^b)$$

$$\mathsf{S}_{pp}{}^{ab}=(\tau\nabla N^a,\nabla N^b)$$

$$\mathsf{S}_{pu_j}{}^{ab}=(\tau\partial_j N^a,\mathbf{u}_h^{n+\theta}\cdot\nabla N^b)$$

$$\mathsf{S}_{py_j}{}^{ab}=(\tau\partial_j N^a,N^b)$$

$$\mathsf{C}(\mathsf{U}^{n+\theta})_{ij}^{ab}=(N^a,\mathbf{u}_h^{n+\theta}\cdot\nabla N^b)\delta_{ij}$$

$$\mathsf{F}_i^a=(N^a,f_i)$$

It is understood that all the arrays are matrices (except F, which is a vector) whose components are obtained by grouping together the left indexes in the previous expressions (a and possibly i) and the right indexes (b and possibly j). Likewise, (21) and (22) need to be modified to account for the Dirichlet boundary conditions (matrix G can be replaced by $-\mathsf{D}^t$ when this is done). Finally, observe that all the matrices of the stabilization terms (denoted by S with a subscript) depend on the velocity. We shall explicitly indicate how when necessary.

4.2. *Fractional step scheme*

As in References [21, 22], we introduce the fractional step scheme at the discrete level. Equation (21) is exactly equivalent to the system

$$\mathsf{M}\frac{1}{\delta t}(\tilde{\mathsf{U}}^{n+1}-\mathsf{U}^n)+\mathsf{K}(\mathsf{U}^{n+\theta})\mathsf{U}^{n+\theta}+\mathsf{G}\mathsf{P}^n+\mathsf{S}_{uu}\mathsf{U}^{n+\theta}+\mathsf{S}_{up}\mathsf{P}^{n+\theta}-\mathsf{S}_{uy}\mathsf{Y}^{n+\theta}=\mathsf{F}^{n+\theta} \tag{24}$$

$$\mathsf{M}\frac{1}{\delta t}(\mathsf{U}^{n+1}-\tilde{\mathsf{U}}^{n+1})+\mathsf{G}(\mathsf{P}^{n+\theta}-\mathsf{P}^n)=0 \tag{25}$$

where $\tilde{\mathsf{U}}^{n+1}$ is an auxiliary variable. If the solution of the discrete problem behaves as we could expect, form (25) we see that the difference between U^{n+1} and $\tilde{\mathsf{U}}^{n+1}$ will be of order $\mathcal{O}(\delta t^2)$. At this point we can make the essential approximation of replacing $\mathsf{U}^{n+\theta}$ by $\tilde{\mathsf{U}}^{n+\theta}:=\theta\tilde{\mathsf{U}}^{n+1}+(1-\theta)\mathsf{U}^n$ in (24) and also in (23). This should not alter the accuracy of the time integration scheme, which is at most $\mathcal{O}(\delta t^2)$. Likewise, we can express $\mathsf{U}^{n+\theta}$ in terms of $\tilde{\mathsf{U}}^{n+\theta}$ using (25) and insert the result in (22), which yields

$$\theta\delta t\mathsf{D}\mathsf{M}^{-1}\mathsf{G}(\mathsf{P}^{n+\theta}-\mathsf{P}^n)-\mathsf{S}_{pp}\mathsf{P}^{n+\theta}-\mathsf{S}_{pu}\mathsf{U}^{n+\theta}+\mathsf{S}_{py}\mathsf{Y}^{n+\theta}-\mathsf{D}\mathsf{U}^{n+\theta}=0 \tag{26}$$

At this point, it is very convenient to make a further approximation. Observe that $\mathsf{D}\mathsf{M}^{-1}\mathsf{G}$ represents an approximation to the Laplacian operator. In order to avoid dealing with this matrix (which is computationally feasible only if M is approximated by a diagonal matrix), we can approximate

$$\mathsf{D}\mathsf{M}^{-1}\mathsf{G}\approx\mathsf{L},\quad \text{with components } \mathsf{L}^{ab}=-(\nabla N^a,\nabla N^b) \tag{27}$$

Matrix L is the standard approximation to the Laplacian operator. Clearly, this approximation is only possible when continuous pressure interpolations are employed.

The terms coming from the stabilization are multiplied by the stabilization parameter τ. From (14) it is seen that it is of order of the critical time step of an explicit scheme (that is, $\theta=0$ everywhere except for the pressure and the incompressibility constraint, which must be treated implicitly). Therefore, if we evaluate the stabilization terms with unknowns at the time level n instead of $n+\theta$ the difference will be of order $\mathcal{O}(\tau\delta t)$. We will do this for the moment, although this approximation can be corrected in the predictor–corrector scheme presented below. In any case, it is shown in References [23, 24] that treating the projection term explicitly has even (slightly) better stability.

After using approximation (27) in (26) and evaluating the unknowns of some of the stabilization terms at time level n, the problem to be solved is

$$\mathbf{M}\frac{1}{\delta t}(\tilde{\mathbf{U}}^{n+1}-\mathbf{U}^{n})+\mathbf{K}(\tilde{\mathbf{U}}^{n+\theta})\tilde{\mathbf{U}}^{n+\theta}+\mathbf{G}\mathbf{P}^{n}$$
$$+\mathbf{S}_{uu}(\tilde{\mathbf{U}}^{n+\theta})\tilde{\mathbf{U}}^{n+\theta}+\mathbf{S}_{up}(\tilde{\mathbf{U}}^{n+\theta})\mathbf{P}^{n}-\mathbf{S}_{uy}(\tilde{\mathbf{U}}^{n+\theta})\mathbf{Y}^{n}=\mathbf{F}^{n+\theta} \tag{28}$$

$$\theta\delta t\mathbf{L}(\mathbf{P}^{n+\theta}-\mathbf{P}^{n})-\mathbf{S}_{pp}(\tilde{\mathbf{U}}^{n+\theta})\mathbf{P}^{n+\theta}-\mathbf{S}_{pu}(\tilde{\mathbf{U}}^{n+\theta})\mathbf{U}^{n}+\mathbf{S}_{py}(\tilde{\mathbf{U}}^{n+\theta})\mathbf{Y}^{n}-\mathbf{D}\tilde{\mathbf{U}}^{n+\theta}=0 \tag{29}$$

$$\mathbf{M}\frac{1}{\delta t}(\mathbf{U}^{n+1}-\tilde{\mathbf{U}}^{n+1})+\mathbf{G}(\mathbf{P}^{n+\theta}-\mathbf{P}^{n})=0 \tag{30}$$

$$\mathbf{M}\mathbf{Y}^{n+\theta}-\mathbf{C}(\tilde{\mathbf{U}}^{n+\theta})\tilde{\mathbf{U}}^{n+\theta}-\mathbf{G}\mathbf{P}^{n+\theta}=0 \tag{31}$$

where the parameter τ is computed with the intermediate velocity $\tilde{\mathbf{U}}^{n+\theta}$. These equations have been written in the order they can be solved: first, one can solve (28) to obtain $\tilde{\mathbf{U}}^{n+1}$, then (29) allows us to obtain $\mathbf{P}^{n+1}$, (30) yields the end-of-step velocity $\mathbf{U}^{n+1}$ and finally (31) yields $\mathbf{Y}^{n+\theta}$.

Problem (28)–(31) is the fractional step version of the stabilized finite element method we propose. Its stability when convection is not stabilized (that is, when the convective term is dropped from the stabilization terms) is analysed in Reference [25]. Its good pressure stability properties come from the term $-\mathbf{S}_{pp}\mathbf{P}^{n+\theta}+\mathbf{S}_{py}\mathbf{Y}^{n}$, which is also present in the original first order fractional step scheme of Chorin [7] and Temam [8]. This was originally identified by Zienkiewicz *et al.*, and exploited in a series of papers [26–29]. In the formulation proposed in these works, applied to both compressible and incompressible flows, convection is dealt with a characteristic based approach, whereas pressure stability relies on a splitting of the equations with a stabilization effect similar to that of scheme (28)–(31). This is why the resulting formulation was termed CBS, standing for *characteristic based split*. A summary of this approach, together with a comparison with the well known Galerkin/least-squares (GLS) stabilization, can be found in Reference [30].

4.3. *Predictor–corrector iterative scheme*

System (28)–(31) is non-linear, and therefore the first step to solve it is to linearize it. Likewise, we have uncoupled some variables appearing in the stabilization terms by treating them explicitly. There is also the possibility of uncoupling these variables by using a block-iterative scheme. In the same iterative loop we can deal with the linearization of the convective term in the momentum equation (28) and the stabilization terms (those multiplied by the parameter τ), although there is of course the possibility to use nested loops.

Denoting by a superscript the iteration counter, the linearized form of system (28)–(31) we propose is

$$\mathbf{M}\frac{1}{\delta t}(\tilde{\mathbf{U}}^{n+1,i+1}-\mathbf{U}^{n})+\mathbf{K}(\tilde{\mathbf{U}}^{n+\theta,i})\tilde{\mathbf{U}}^{n+\theta,i+1}+\mathbf{G}\mathbf{P}^{n}$$
$$+\mathbf{S}_{uu}(\tilde{\mathbf{U}}^{n+\theta,i})\tilde{\mathbf{U}}^{n+\theta,i+1}+\mathbf{S}_{up}(\tilde{\mathbf{U}}^{n+\theta,i})\mathbf{P}^{n}-\mathbf{S}_{uy}(\tilde{\mathbf{U}}^{n+\theta,i})\mathbf{Y}^{n}=\mathbf{F}^{n+\theta} \tag{32}$$

$$\theta\delta t\mathsf{L}(\mathsf{P}^{n+\theta,i+1}-\mathsf{P}^{n})-\mathsf{S}_{pp}(\tilde{\mathsf{U}}^{n+\theta,i+1})\mathsf{P}^{n+\theta,i+1}-\mathsf{S}_{pu}(\tilde{\mathsf{U}}^{n+\theta,i+1})\mathsf{U}^{n}$$
$$+\mathsf{S}_{py}(\tilde{\mathsf{U}}^{n+\theta,i+1})\mathsf{Y}^{n}-\mathsf{D}\tilde{\mathsf{U}}^{n+\theta,i+1}=0 \tag{33}$$

$$\mathsf{M}\frac{1}{\delta t}(\mathsf{U}^{n+1,i+1}-\tilde{\mathsf{U}}^{n+1,i+1})+\mathsf{G}(\mathsf{P}^{n+\theta,i+1}-\mathsf{P}^{n})=0 \tag{34}$$

$$\mathsf{M}\mathsf{Y}^{n+\theta,i+1}-\mathsf{C}(\tilde{\mathsf{U}}^{n+\theta,i+1})\tilde{\mathsf{U}}^{n+\theta,i+1}-\mathsf{G}\mathsf{P}^{n+\theta,i+1}=0 \tag{35}$$

These equations are all linear and uncoupled, that is to say, they can be solved successively.

All the arguments that led us to the fractional step scheme (28)–(31) are valid if instead of using the pressure P^n we replace it by any other pressure. In particular, in the iterative scheme (32)–(35) we can replace it by $\mathsf{P}^{n+\theta,i}$. If the resulting iterative scheme converges, the second term in the left-hand side of (34) will disappear, and therefore *the intermediate velocity will converge to the end-of-step one*. Thus, we do not need to distinguish between $\tilde{\mathsf{U}}$ and U and (34) can be simply ignored. The final iterative scheme is

$$\mathsf{M}\frac{1}{\delta t}(\mathsf{U}^{n+1,i+1}-\mathsf{U}^{n})+\mathsf{K}(\mathsf{U}^{n+\theta,i})\mathsf{U}^{n+\theta,i+1}+\mathsf{G}\mathsf{P}^{n+\theta,i}$$
$$+\mathsf{S}_{uu}(\mathsf{U}^{n+\theta,i})\mathsf{U}^{n+\theta,i+1}+\mathsf{S}_{up}(\mathsf{U}^{n+\theta,i})\mathsf{P}^{n+\theta,i}-\mathsf{S}_{uy}(\mathsf{U}^{n+\theta,i})\mathsf{Y}^{n+\theta,i}=\mathsf{F}^{n+\theta} \tag{36}$$

$$\theta\delta t\mathsf{L}(\mathsf{P}^{n+\theta,i+1}-\mathsf{P}^{n+\theta,i})-\mathsf{S}_{pp}(\mathsf{U}^{n+\theta,i+1})\mathsf{P}^{n+\theta,i+1}-\mathsf{S}_{pu}(\mathsf{U}^{n+\theta,i+1})\mathsf{U}^{n+\theta,i+1}$$
$$+\mathsf{S}_{py}(\mathsf{U}^{n+\theta,i+1})\mathsf{Y}^{n+\theta,i}-\mathsf{D}\mathsf{U}^{n+\theta,i+1}=0 \tag{37}$$

$$\mathsf{M}\mathsf{Y}^{n+\theta,i+1}-\mathsf{C}(\mathsf{U}^{n+\theta,i+1})\mathsf{U}^{n+\theta,i+1}-\mathsf{G}\mathsf{P}^{n+\theta,i+1}=0 \tag{38}$$

Apparently, this is a straightforward iteration procedure for solving the original *monolithic* problem (21)–(23) freezing the pressure gradient and the projection in the momentum equation. However, there is a term whose presence would be hardly motivated by looking only at this system, namely, the term $\theta\delta t\mathsf{L}(\mathsf{P}^{n+\theta,i+1}-\mathsf{P}^{n+\theta,i})$. The motivation to introduce it comes from the inspection of what happens in the fractional step scheme we have described, even though now we aim to converge to the original non-split problem (21)–(23).

5. NODAL BASED IMPLEMENTATION

5.1. Motivation

The objective of this section is to summarize the nodal based implementation proposed in Reference [16] and to describe how to apply it to the predictor–corrector scheme of the previous section.

The matrix of the final algebraic system changes from time step to time step and from iteration to iteration due to its dependence with the velocity. The latter could be avoided by

treating these terms explicitly in time, but this would be at the expense of loosing stability of the time integration.

The time consuming task in the calculation of the matrix of the algebraic system (traditionally referred to as 'stiffness matrix') is the numerical integration involved. However, *it is possible to introduce some approximations that allow to express all the integrals in terms of*

$$\int_\Omega N^a N^b \,\mathrm{d}\Omega$$

$$\int_\Omega N^a \partial_i N^b \,\mathrm{d}\Omega, \quad \int_\Omega \partial_i N^a N^b \,\mathrm{d}\Omega, \ i = 1, \ldots, n_{\mathrm{sd}} \tag{39}$$

$$\int_\Omega \partial_i N^a \partial_j N^b \,\mathrm{d}\Omega, \quad i, j = 1, \ldots, n_{\mathrm{sd}}$$

for $a, b = 1, \ldots, n_{\mathrm{pts}}$, the total number of nodal points. For fixed domains Ω, *all the integrals in* (39) *can be computed at the beginning of the run and stored.*

At this point there are two questions to be treated. The first is which are the approximations needed to be able to use only (39) to build up the matrix of the algebraic system. This is the subject of Sections 5.2 and 5.3.

The second question is how to store the integrals in (39). The efficiency of the overall implementation depends on how efficient the storage scheme is. The method we employ is described in Section 5.4.

5.2. Approximation of the convective term

All the terms coming from a *bilinear* form can be obviously computed using the integrals in (39), but approximations will be required for the nonlinear terms. We discuss here the approximation of the convective term. When the viscous term is nonlinear (either due to the constitutive behaviour or to turbulence modelling) the approximations required are described in Reference [16].

Let us begin with the convective term in (36), which comes from the term $(\mathbf{u}_h^{n+\theta} \cdot \nabla \mathbf{u}_h^{n+\theta}, \mathbf{v}_h)$ in the linearized version of the discrete variational problem (18).

Let $\mathbf{a}_h \equiv \mathbf{u}_h^{n+\theta,i}$ and $\mathbf{u}_h \equiv \mathbf{u}_h^{n+\theta,i+1}$. When the velocity test function is taken such that $v_{h,i} = N^a \delta_{ik}$, with k fixed ($k = 1, \ldots, n_{\mathrm{sd}}$), the convective term is

$$\int_\Omega \mathbf{v}_h \cdot (\mathbf{a}_h \cdot \nabla \mathbf{u}_h) \,\mathrm{d}\Omega = \sum_{j=1}^{n_{\mathrm{sd}}} \left(\int_\Omega N^a a_{h,j} \partial_j N^b \,\mathrm{d}\Omega \right) U_k^b \tag{40}$$

The need for an additional approximation arises because of the function $a_{h,j}$ appearing within the integrals. Calling A_j^a the nodal values of this function, the approximation that is proposed here is

$$\sum_{j=1}^{n_{\mathrm{sd}}} \left(\int_\Omega N^a a_{h,j} \partial_j N^b \,\mathrm{d}\Omega \right) \approx \sum_{j=1}^{n_{\mathrm{sd}}} A_j^c \left(\int_\Omega N^a \partial_j N^b \,\mathrm{d}\Omega \right) \tag{41}$$

where $c = b$ or a. In any case, the convective term will be expressed in terms of the integrals of (39), as desired.

The reasons for choice $c=a$ are explained in Reference [16]. We prefer the choice $c=b$, which can be justified as follows. Since $\mathbf{a}_h$ is approximately divergence free, we can approximate

$$\int_\Omega \mathbf{v}_h \cdot (\mathbf{a}_h \cdot \nabla \mathbf{u}_h)\, \mathrm{d}\Omega \approx \int_\Omega \mathbf{v}_h \cdot [\nabla \cdot (\mathbf{a}_h \otimes \mathbf{u}_h)]\, \mathrm{d}\Omega \tag{42}$$

In fact, it is not necessary to consider this as an approximation, since the convective term of the original continuous equations could have been written directly as $\nabla \cdot (\mathbf{u} \otimes \mathbf{u})$ rather than $\mathbf{u} \cdot \nabla \mathbf{u}$. What is definitely an approximation is *to interpolate the product* $\mathbf{a}_h \otimes \mathbf{u}_h$ *instead of each of the components separately*. Doing this when $v_{h,i} = N^a \delta_{ik}$ yields

$$\begin{aligned} \sum_{i,j=1}^{n_{\mathrm{sd}}} \int_\Omega v_{h,i} \partial_j (a_{h,j} u_{h,i})\, \mathrm{d}\Omega &\approx \sum_{j=1}^{n_{\mathrm{sd}}} \int_\Omega N^a \partial_j \left(\sum_{a=1}^{n_{\mathrm{pts}}} N^a A_j^b U_k^b \right) \mathrm{d}\Omega \\ &= \sum_{j=1}^{n_{\mathrm{sd}}} \sum_{b=1}^{n_{\mathrm{pts}}} A_j^b \left(\int_\Omega N^a \partial_j N^b\, \mathrm{d}\Omega \right) U_k^b \end{aligned} \tag{43}$$

which justifies (41) for $c=b$.

5.3. Nodal stabilization parameters: Consistency and conservation

The last point that needs to be analysed is the way in which the stabilization terms appearing in (36)–(38) can be approximated to achieve the goal of using only the integrals in (39) in the implementation. For the purposes of this section, it suffices to consider the stationary version of problem (18) and (19).

As before, let $\mathbf{a}_h$ be the velocity of the previous iteration, $\mathbf{u}_h$ the velocity field that needs to be computed and $\mathbf{y}_h$ the projection of the convective and pressure gradient terms from a known iteration. The discrete problem to be solved can be written as

$$\begin{aligned} &\int_\Omega \mathbf{v}_h \cdot (\mathbf{a}_h \cdot \nabla \mathbf{u}_h)\, \mathrm{d}\Omega + \nu \int_\Omega \nabla \mathbf{v}_h : \nabla \mathbf{u}_h\, \mathrm{d}\Omega - \int_\Omega p_h \nabla \cdot \mathbf{v}_h\, \mathrm{d}\Omega \\ &\quad + S_{\mathrm{mom}}(\mathbf{v}_h; \mathbf{u}_h, p_h) = R_{\mathrm{mom}}(\mathbf{v}_h) \end{aligned} \tag{44}$$

$$\int_\Omega q_h \nabla \cdot \mathbf{u}_h + S_{\mathrm{cont}}(q_h; \mathbf{u}_h, p_h) = R_{\mathrm{cont}}(q_h) \tag{45}$$

where the right-hand side terms $R_{\mathrm{mom}}(\mathbf{v}_h)$ and $R_{\mathrm{cont}}(q_h)$ are

$$R_{\mathrm{mom}}(\mathbf{v}_h) := \int_\Omega \mathbf{v}_h \cdot \mathbf{f}\, \mathrm{d}\Omega + \int_\Omega \tau (\mathbf{a}_h \cdot \nabla \mathbf{v}_h) \cdot \mathbf{y}_h\, \mathrm{d}\Omega$$

$$R_{\mathrm{cont}}(q_h) := \int_\Omega \tau \nabla q_h \cdot \mathbf{y}_h\, \mathrm{d}\Omega$$

and the stabilization terms are given by

$$S_{\text{mom}}(\mathbf{v}_h; \mathbf{u}_h, p_h) = \int_\Omega \tau(\mathbf{a}_h \cdot \nabla \mathbf{v}_h) \cdot (\mathbf{a}_h \cdot \nabla \mathbf{u}_h + \nabla p_h)\, \mathrm{d}\Omega \tag{46}$$

$$S_{\text{cont}}(q_h; \mathbf{u}_h, p_h) = \int_\Omega \tau \nabla q_h \cdot (\mathbf{a}_h \cdot \nabla \mathbf{u}_h + \nabla p_h)\, \mathrm{d}\Omega \tag{47}$$

where the stabilization parameter τ is given by (14).

It is explained in Reference [16] and also in Reference [31] that *a sufficient conditions for the stabilized finite element method to be globally conservative* is that

$$\sum_{a=1}^{n_{\text{pts}}} S_{\text{mom}}(N^a \mathbf{e}_k; \mathbf{u}_h, p_h) = 0 \tag{48}$$

$$\sum_{a=1}^{n_{\text{pts}}} S_{\text{cont}}(N^a; \mathbf{u}_h, p_h) = 0 \tag{49}$$

where $\mathbf{e}_k$ is the unit vector along the x_k coordinate. In a standard implementation of the formulation, the stabilization parameters are computed depending only on the element, and not on the test functions being used in (46) and (47). However, the idea of the nodal-based implementation presented here is to use nodal values for all the parameters of the formulation, and in particular for τ. It can be readily observed from (46) and (47) for $\mathbf{v}_h = N^a \mathbf{e}_k$ that conditions (48) and (49) *will not hold if τ depends on a*. If this happens, it is impossible to assess that the numerical formulation is conservative.

Despite this lack of 'conservation', which has to be acknowledged, the stabilization parameter τ will be evaluated at node a when $\mathbf{v}_h = N^a \mathbf{e}_k$ and when $q_h = N^a$. The reason for this is related to the consistency of the scheme, which is discussed next. We have to mention that we have observed no consequences of this lack of conservation for the case of laminar incompressible flows considered in this paper.

Let us consider the stabilization term (46). Taking $\mathbf{v}_h = N^a \mathbf{e}_k$ and interpolating the velocity and the pressure it is found that

$$S_{\text{mom}}(N^a \mathbf{e}_k; \mathbf{u}_h, p_h) = \sum_{b=1}^{n_{\text{pts}}} \left[\sum_{i,j=1}^{n_{\text{sd}}} \int_\Omega \tau (a_{h,i} \partial_i N^a)(a_{h,j} \partial_j N^b)\, \mathrm{d}\Omega \right] U_k^b + \sum_{b=1}^{n_{\text{pts}}} \left[\sum_{i=1}^{n_{\text{sd}}} \int_\Omega \tau (a_{h,i} \partial_i N^a) \partial_k N^b\, \mathrm{d}\Omega \right] P^b \tag{50}$$

The goal now is to make the convenient approximations to write this in terms of the integrals in (39). First of all, observe that the integrals involving the shape functions of nodes a and b can be extended over $\Omega_a \cap \Omega_b$ only, the intersection of the interior of the supports of N^a and N^b. The basic idea is that the velocity and *the stabilization parameter τ will be evaluated either at node a or at node b, or a combination of both*. However, it is not a

matter of choice. The only possibility to approximate (50) is to take:

- The velocity $\mathbf{a}_h$ coming from the element residual of the differential equation evaluated at node b, and thus the ith component equal to the nodal value A_i^b. This corresponds to the approximation of the convective term discussed in the previous section.
- The velocity $\mathbf{a}_h$ appearing in the operator applied to the test function evaluated at node a, and thus the ith component equal to the nodal value A_i^a. *This is needed for consistency reasons*: *if* $\mathbf{a}_h$ *depends also on node* b, *exact nodal values of the velocity would not satisfy the discrete variational equations*. The reason is that the sum over b in (50) would not produce the convective and pressure gradient terms $\mathbf{a}_h \cdot \mathbf{u}_h + \nabla p_h$ if the partial derivatives of N^b are multiplied by a factor that depends on node b. However, the same discussion concerning the choice of the stabilization parameters is applicable now: the resulting numerical scheme *will not* be conservative.
- For the same reason as before, the parameter τ needs to be evaluated at node a. Even though this produces a non-conservative scheme, it is essential to have a consistent numerical method, in the sense that exact solutions of the continuous problem should also be solutions of the discrete one, provided they belong to the finite element space. The expressions we use for this parameter is (14), taking $\mathbf{u}_h$ as the nodal velocity at node a and h as the minimum distance from node a to the surrounding nodes.

Using all these approximations in (50) one finds

$$S_{\mathrm{mom}}(N^a \mathbf{e}_k; \mathbf{u}_h, p_h) \approx \sum_{b=1}^{n_{\mathrm{pts}}} \left[\sum_{i,j=1}^{n_{\mathrm{sd}}} A_i^a A_j^b \tau^a \left(\int_\Omega \partial_i N^a \partial_j N^b \, \mathrm{d}\Omega \right) \right] U_k^b + \sum_{b=1}^{n_{\mathrm{pts}}} \left[\sum_{i=1}^{n_{\mathrm{sd}}} A_i^a \tau^a \left(\int_\Omega \partial_i N^a \partial_k N^b \, \mathrm{d}\Omega \right) \right] P^b$$

Once again, the objective of using only the integrals in (39) has been accomplished.

Similarly, the final approximation of the stabilization term in the continuity equation when the test function is taken $q_h = N^a$ will be

$$S_{\mathrm{cont}}(N^a; \mathbf{u}_h, p_h) \approx \sum_{b=1}^{n_{\mathrm{pts}}} \left[\sum_{i,j=1}^{n_{\mathrm{sd}}} \tau^a A_j^b \left(\int_\Omega \partial_i N^a \partial_j N^b \, \mathrm{d}\Omega \right) \right] U_k^b + \sum_{b=1}^{n_{\mathrm{pts}}} \left[\sum_{i=1}^{n_{\mathrm{sd}}} \tau^a \left(\int_\Omega \partial_i N^a \partial_i N^b \right) \right] P^b \tag{51}$$

There is a very important remark to be made here. The fact that τ is evaluated at node a in the last term of (51) *makes this non-symmetric with respect to a and b*. This has important consequences, since this is the term which precisely gives matrix $\mathsf{S}_{pp}(\tilde{\mathsf{U}}^{n+\theta,i+1})$ in (33) (the dependence on the velocity comes through the stabilization parameter). Therefore, this matrix will not be symmetric, and it will not be possible to solve (33) using a method to solve linear systems with symmetric matrices, such as conjugate gradient in the present case. In

practice, the option that we have found most effective is *to take τ constant for all the nodes in the continuity equation*. In particular, we take this parameter equal to the *minimum* of the values computed for all the nodes. We have observed that the precise value of τ has not much influence in the quality of the results, but it does in the convergence rate of the conjugate gradient scheme. Apparently, taking it as the minimum for all the nodes is what yields a best conditioned matrix.

5.4. Mesh graph and basic algorithm

Once it has been established that the matrix of the algebraic system can be built up making use of the integrals in (39), the question that needs to be addressed is how to store these integrals in an efficient way. The technique adopted in this work is to use a *compressed sparse row* (CSR) format to store the $n_{\text{pts}} \times n_{\text{pts}}$ 'virtual' matrix $\mathbf{M}_{\text{mesh}}$, whose coefficients are $M^{ab}_{\text{mesh}} = 1$ if nodes a and b are connected, $= 0$ otherwise. This is the matrix of the graph associated to the finite element mesh.

In order to store $\mathbf{M}_{\text{mesh}}$ using the CSR format, two arrays are needed. Let NZD be the number of non-zero coefficients in $\mathbf{M}_{\text{mesh}}$ These two arrays are

$$R_{\text{mesh}}(n_{\text{pts}}); \quad R_{\text{mesh}}(a) = \text{Component of } \mathbf{M}_{\text{mesh}} \text{ where row number } a \text{ starts} \tag{52}$$

$$C_{\text{mesh}}(\text{NZD}); \quad C_{\text{mesh}}(I) = \text{Column in } \mathbf{M}_{\text{mesh}} \text{ of the component } I \tag{53}$$

For implementation convenience, it is useful to take R_{mesh} of dimension $n_{\text{pts}} + 1$, with $R_{\text{mesh}}(n_{\text{pts}} + 1) = \text{NZD} + 1$ (see the algorithm of Box 1).

The arrays in (52) and (53) allow to access to all the components of the integrals in (39) when these are stored in $n_{\text{pts}} \times n_{\text{pts}}$ matrices (each component of which can be a n_{sd} vector or a $n_{\text{sd}} \times n_{\text{sd}}$ matrix). Moreover, $R_{\text{mesh}}(n_{\text{pts}})$ and $C_{\text{mesh}}(\text{NZD})$ can also be used to store in CSR format the stiffness matrix of the problem, and thus they define completely its topology. Once they have been computed, the memory for the classical array of nodal connections, which has the list of nodes that each element has, can be freed.

To compute the contributions to the force vector due to the surface traction when it exists, the integrals of the shape functions products over the boundaries are needed. However, these can be obtained from the corresponding integrals over the interior of the domain using the expression

$$\int_{\partial\Omega} n_i N^a N^b \, \mathrm{d}\Gamma = \int_{\Omega} \partial_i N^a N^b \, \mathrm{d}\Omega + \int_{\Omega} N^a \partial_i N^b \, \mathrm{d}\Omega, \quad i = 1, \ldots, n_{\text{sd}} \tag{54}$$

where n_i is the ith component of the exterior normal to $\partial\Omega$. The left-hand side of (54) is equal to $n_i(\bar{\mathbf{x}})$ multiplied by the integral of $N^a N^b$, where $\bar{\mathbf{x}}$ is a point in $\partial\Omega \cap \bar{\Omega}_a \cap \bar{\Omega}_b$. Assuming that this point is the same for all i it is found that

$$\int_{\partial\Omega} N^a N^b \, \mathrm{d}\Gamma \approx \pm \left[\sum_{i=1}^{n_{\text{sd}}} \left(\int_{\partial\Omega} n_i N^a N^b \, \mathrm{d}\Gamma \right)^2 \right]^{1/2} \tag{55}$$

The sign of this integral is determined from the sign of the integrals in (54).

To store the boundary integrals in (55) it is a waste of memory to use a matrix with the same sparsivity pattern as $\mathbf{M}_{\text{mesh}}$. It is preferable to store also the boundary graph, that is,

the matrix $\mathbf{M}_{\text{boun}}$, of dimensions $n_{\text{bpt}} \times n_{\text{bpt}}$, where n_{bpt} is the number of boundary nodes. If NZB is the number of non-zero coefficients of $\mathbf{M}_{\text{boun}}$, the arrays needed to store it are

$$R_{\text{boun}}(n_{\text{bpt}}); \quad R_{\text{boun}}(a) = \text{Component of } \mathbf{M}_{\text{boun}} \text{ where row number } a \text{ starts} \tag{56}$$

$$C_{\text{boun}}(NZB); \quad C_{\text{boun}}(I) = \text{Column in } \mathbf{M}_{\text{boun}} \text{ of the component } I \tag{57}$$

As for R_{mesh}, it is useful to take R_{boun} of dimension $n_{\text{bpt}} + 1$ and $R_{\text{boun}}(n_{\text{bpt}} + 1) = NZB + 1$.

Let us see now how the arrays in (52) and (53) (and (56) and (57)) allow to construct the stiffness matrix and force vector of the algebraic problem using the approximations described in the previous section.

The arrays defined in Section 4.1 can be computed by looping first over each nodal point and then over the nodes connected to it, as indicated in Box 1. This can be done making use of (52) and (53). This algorithm turns out to be very efficient compared to the standard loop over the elements to compute the element contributions. In particular, all gather–scatter operations are avoided, and there is no need to perform the classical assembly operations.

Box 1. Construction of the arrays of the algebraic system

For $a = 1, n_{\text{pts}}$ DO:

- Compute τ^a
- Set the force vectors to zero

FOR $I = R_{\text{mesh}}(a), R_{\text{mesh}}(a+1) - 1$ DO:

- Identify the column node: $b = C_{\text{mesh}}(I)$
- Compute the contributions to the matrix components ab of the matrices defined in Section 4.1
- Add contribution to the force vectors

END

END

6. NUMERICAL EXAMPLES

This section presents a couple of numerical examples to illustrate, first, how nodal-based fractional step and nodal-based monolithic schemes lead to nearly identical results and, second, how the gains of nodal-based formulations in matrix construction become important in real 3D simulations.

The first example is the classical two-dimensional driven cavity flow benchmark at Reynolds 1000. In this test, a fluid is confined within a square cavity in which the upper edge slides tangentially to induce the fluid motion. The problem has been solved in time using a first-order integration scheme until a convergence of 10^{-4} in the Euclidian norm of velocity increments has been achieved. The time step has been taken as $\delta t = 0.1$. A total of only 141 time steps has been needed to reach the steady state. The GMRES has been used to solve nonsymmetric

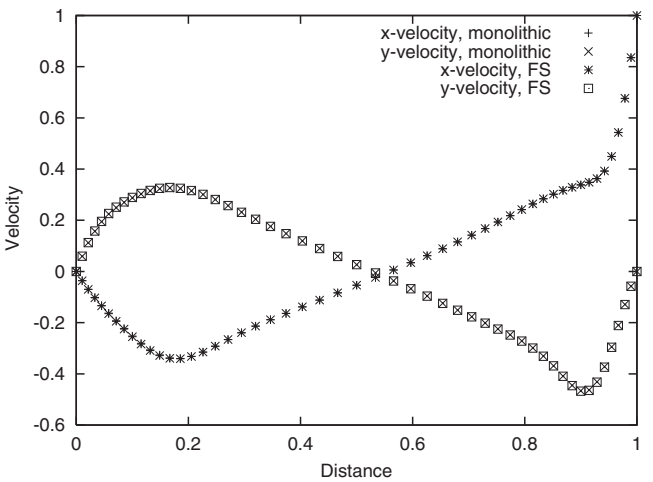

Figure 1. Velocity profiles along vertical and horizontal sections for a cavity flow at Reynolds 1000. Results using a mesh with 2500 Q1 elements. No differences appear between the nodal-based monolithic and predictor–corrector schemes.

Table I. Comparison of CPU times for the cavity flow problem. The solver CPU corresponds to 10 GMRES iterations for the monolithic scheme and 10 iterations of GMRES for the velocity components and of conjugate gradient (with ILU preconditioning) for the pressure in the case of the predictor–corrector scheme.

Method	Matrix CPU (s) Total/Ratio	Solver CPU (s) Total/Ratio
Monolithic NB	0.134/0.77	0.076/2.4
Predictor–corrector NB	0.175/1.0	0.032/1.0

problems, whereas the symmetric problem (33) has been solved using a conjugate gradient method with ILU preconditioning. *The same number of iterations per time step has been required using the monolithic* (M) *scheme and the predictor–corrector* (PC) *one*. This means that the iterative coupling of the velocity and the pressure can be achieved within the same iterative loop as the non-linearity of the problem.

Figure 1 shows the steady velocity profiles along vertical and horizontal lines through the geometric center of the cavity for both the nodal-based monolithic scheme and the nodal-based predictor–corrector scheme. Both results are identical, which demonstrates that the converged solution of the PC scheme is certainly that of the monolithic method. Otherwise, there would be an error of order $\mathcal{O}(\delta t^2)$ due to the splitting of the equations (in this case, δt is large, compared to the critical time step of the problem).

Table I gives the CPU time per iteration spent in matrix construction and solution of the resulting algebraic system, referring the results to the PC scheme and giving the ratio of the monolithic method with respect to the former. Surprisingly, it is observed that PC requires always between 15 and 30% more of CPU time to build the matrix. Note that this is a characteristic feature of the nodal implementation that contrasts with the element based one, where fractional step methods require always less CPU time for matrix construction than

Table II. Comparison of CPU times and memory requirements for different methods for the aerodynamic analysis of a market place.

Method	Matrix CPU (s) Total/Ratio	Solver CPU (s) Total/Ratio	Memoy (Mb) Total/Ratio
Monolithic EB, 1 pt	530.6/7.94	—/—	343/0.82
Monolithic EB, 4 pt	839.4/12.6	—/—	1020/2.44
Monolithic NB	47.2/0.71	47.5/3.4	—/—
Predictor–corrector NB	66.8/1.0	14.0/1.0	418/1.0

monolithic ones. However, the difference is small compared to the difference between nodal- and element-based, as we shall see in the following example. This increase in the amount of CPU is due to the way the calculations are carried out: in the case of the monolithic scheme, all the terms can be computed with a single loop as the one indicated in Box 1, whereas for the PC method several loops are required, since the velocity components and the pressure are solved one after the other.

The second example is an aerodynamic study of a market place. This is an example of a real simulation case, included here to demonstrate the feasibility of the nodal based implementation and the comparison of the monolithic and PC schemes. The finite element mesh consists of approximately 1.5 million linear tetrahedral elements.

The goal of this simulation was to compute the air flow within a market place with different architectural situations and compute variables related to people's comfort, such as wind velocity, temperature and chilling factor. Plates 1 and 2 are included only to understand the setting of the problem. Our objective here is only to compare the performance of the monolithic and PC schemes with a nodal based implementation and also with a classical element based implementation of the monolithic scheme. Table II summarizes the results. Again, results are referred to the PC method, and the ratios of the other schemes with respect to this one are given. In this case, the flow is not stationary. A more or less uniform spectrum is reached after approximately one thousand time steps with δt computed as 100 times the critical time step of the explicit Euler scheme.

The first column gives the CPU time needed to construct the matrices of the final algebraic system in one iteration of a time step. It is interesting to note that, again, the monolithic method does this slightly faster. What is really important is the difference with the element based implementation. In this case, results depend on how many integration points are used. Both one point and four points per element have been considered. The results obtained are similar to those presented in Reference [16] for a test problem.

The second column of Table II compares the CPU required to solve the linear systems of one iteration of a time step for the nodal based formulation using the monolithic and the PC schemes (for the element based, results should be identical to those of the monolithic case, since the matrix once constructed is virtually the same in both cases). The gain in CPU using the PC scheme is really important. As in the previous example, the GMRES has been used to solve problems with nonsymmetric matrices and the symmetric problem (33) has been solved using a conjugate gradient method with ILU preconditioning. As explained in Section 5, the stabilization parameter τ has been taken constant to make this problem symmetric.

Concerning the memory requirements, the third column of Table II compares the memory required to store the integrals needed in the nodal based implementation with a standard data

base that our code uses for the element based one. This data base contains basically the Cartesian derivatives of the shape functions and the element volumes at the integration points for each element. Therefore, it is obvious that it depends on the number of integration points employed. It is observed that if only one point is used, the element based implementation is less memory consuming than the nodal based one, but the situation is clearly reversed in the case of four integration points. Observe that in the nodal based approach the integrals can be computed with all the integration points desired, since once they are stored all the element information can be removed from the code.

7. CONCLUSIONS

In this paper, we have discussed several aspects of a finite element formulation to solve incompressible flow problems, such as the application of the stabilization with orthogonal subscales proposed in Reference [17] and the nodal based implementation proposed in Reference [16] used together with a predictor–corrector scheme motivated by a fractional step method.

The emphasis of this paper has not been to assess the quality of the results obtained with the stabilization or the nodal based implementation, but rather to study the aspects of their application to the predictor–corrector proposed in this work. With respect to this point, the main conclusions that can be drawn from the examples of this paper (and corroborated by many other applications not presented here) are the following:

- It is possible to converge to the solution of the monolithic scheme using an iterative method that *uncouples the calculation of the velocity and the pressure*. The robustness of this scheme relies on the term $\theta\delta t\mathsf{L}(\mathsf{P}^{n+\theta,i+1}-\mathsf{P}^{n+\theta,i})$ in (37), whose introduction is motivated by the inspection of a fractional step scheme.
- The uncoupling of the velocity and the pressure allows to reduce drastically the computing time to solve the linear systems of equations that appear in the problem. All the problems to be solved are *scalar*.
- As it was already observed in Reference [16], the nodal based implementation is extremely effective to reduce the computing time to construct the matrices of the problem. However, contrary to what happens in a standard element based approach, *this construction is* (*slightly*) *more expensive using the predictor–corrector* (*or the original fractional step*) *method than the monolithic one*. The reason is that approximately the same number of matrix components are computed, and for the monolithic case the calculations can be organized in a more efficient manner, since only one loop over the nodes and its neighbours is needed.
- Using the nodal-based implementation, the memory requirements of the monolithic and the predictor–corrector methods are the same, and smaller than the memory needed for the element based implementation if exact integration is required.
- The fact that the nodal stabilization parameters have to be evaluated at node b when computing the contribution to the component ba of the matrices has a drawback: it makes matrix S_{pp} in (37) non-symmetric, and therefore methods such as conjugate gradient to solve this algebraic problem cannot be used. We have overcome this problem by using a constant value of τ, the minimum of the values of this parameter computed for all the nodes. In practice, this option has been found to be very effective.

Summarizing, we believe that the model proposed here, both regarding the formulation and the implementation aspects, can be an effective tool for the finite element simulation of real life incompressible flow problems.

REFERENCES

1. Brezzi F, Douglas J. Stabilized mixed methods for the Stokes problem. *Numerische Mathematik* 1988; **53**:225–235.
2. Douglas J, Wang J. An absolutely stabilized finite element method for the Stokes problem. *Mathematics of Computation* 1989; **52**:495–508.
3. Hughes TJR, Franca LP, Balestra M. A new finite element formulation for computational fluid dynamics: V. Circumventing the Babuška–Brezzi condition: a stable Petrov–Galerkin formulation for the Stokes problem accommodating equal-order interpolations. *Computer Methods in Applied Mechanics and Engineering* 1986; **59**:85–99.
4. Franca LP, Hughes TJR. Convergence analyses of Galerkin least-squares methods for advective-diffusive forms of the Stokes and incompressible Navier–Stokes equations. *Computer Methods in Applied Mechanics and Engineering* 1993; **105**:285–298.
5. Franca L, Stenberg R. Error analysis of some Galerkin least-squares methods for the elasticity equations. *SIAM Journal on Numerical Analysis* 1991; **28**:1680–1697.
6. Bochev P, Cai Z, Manteuffel TA, McCormick SF. Analysis of velocity-flux first-order system least-squares principles for the Navier–Stokes equations: part I. *SIAM Journal on Numerical Analysis* 1998; **35**:990–1009.
7. Chorin AJ. A numerical method for solving incompressible viscous problems. *Journal of Computational Physics* 1967; **2**:12–26.
8. Temam R. Sur l'approximation de la solution des équations de Navier–Stokes par la méthode des pas fractionaires (I). *Archives for Rational Mechanics and Analysis* 1969; **32**:135–153.
9. Natarajan R. A numerical method for incompressible viscous flow simulation. *Journal of Computational Physics* 1992; **100**:384–395.
10. Shen J. Hopf bifurcation of the unsteady regularized driven cavity-flow. *Journal of Computational Physics* 1991; **95**:228–245.
11. Turek S. A comparative study of time-stepping techniques for the incompressible Navier–Stokes equations: from fully implicit nonlinear schemes to semi-implicit projection methods. *International Journal for Numerical Methods in Fluids* 1996; **22**:987–1011.
12. Blasco J, Codina R, Huerta A. A fractional step method for the incompressible Navier–Stokes equations related to a predictor-multicorrector algorithm. *International Journal for Numerical Methods in Fluids* 1998; **28**:1391–1419.
13. Codina R. Soto O. Approximation of the incompressible Navier–Stokes equations using orthogonal-subscale stabilization and pressure segregation on anisotropic finite element meshes. *Computer Methods in Applied Mechanics and Engineering*, to appear.
14. Hughes TJR. Multiscale phenomena: Green's function, the Dirichlet-to-Neumann formulation, subgrid scale models, bubbles and the origins of stabilized formulations. *Computer Methods in Applied Mechanics and Engineering* 1995; **127**:387–401.
15. Hughes TJR, Feijóo GR, Mazzei L, Quincy JB. The variational multiscale method—a paradigm for computational mechanics. *Computer Methods in Applied Mechanics and Engineering* 1998; **166**:3–24.
16. Codina R. A nodal-based implementation of a stabilized finite element method for incompressible flow problems. *International Journal for Numerical Methods in Fluids* 2000; **33**:737–766.
17. Codina R. Stabilized finite element approximation of transient incompressible flows using orthogonal subscales. *Computer Methods in Applied Mechanics and Engineering* 2002; **191**:4295–4321.
18. Baiocchi C, Brezzi F, Franca LP. Virtual bubbles and Galerkin/least-squares type methods (Ga.L.S). *Computer Methods in Applied Mechanics and Engineering* 1993; **105**:125–141.
19. Brezzi F, Franca LP, Hughes TJR, Russo A. $b = \int g$. *Computer Methods in Applied Mechanics and Engineering* 1997; **145**:329–339.
20. Codina R. A stabilized finite element method for generalized stationary incompressible flows. *Computer Methods in Applied Mechanics and Engineering* 2001; **190**:2681–2706.
21. Perot JB. An analysis of the fractional step method. *Journal of Computational Physics* 1993; **108**:51–58.
22. Quarteroni A, Saleri F, Veneziani A. Factorization methods for the numerical approximation of Navier–Stokes equations. *Computer Methods in Applied Mechanics and Engineering* 2000; **188**:505–526.
23. Codina R, Blasco J. Stabilized finite element method for the transient Navier–Stokes equations based on a pressure gradient projection. *Computer Methods in Applied Mechanics and Engineering* 2000; **182**:277–300.
24. Blasco J, Codina R. Space and time error estimates for a first order, pressure stabilized finite element method for the incompressible Navier–Stokes equations. *Applied Numerical Mathematics* 2001; **38**:475–497.

25. Codina R. Pressure stability in fractional step finite element methods for incompressible flows. *Journal of Computational Physics* 2001; **170**:112–140.
26. Zienkiewicz OC, Codina R. A general algorithm for compressible and incompressible flow—Part I. The split, characteristic-based scheme. *International Journal for Numerical Methods in Fluids* 1995; **20**:869–885.
27. Zienkiewicz OC, Morgan K, Satya Sai BVK, Codina R, Vázquez M. A general algorithm for compressible and incompressible flow—Part II. Tests on the explicit form. *International Journal for Numerical Methods in Fluids* 1995; **20**:887–913.
28. Codina R, Vázquez M, Zienkiewicz OC. A general algorithm for compressible and incompressible flow—Part III. The semi-implicit form. *International Journal for Numerical Methods in Fluids* 1998; **27**:13–32.
29. Zienkiewicz OC, Nithiarasu P, Codina R, Vázquez M, Ortiz P. The characteristic based split procedure: An efficient and accurate algorithm for fluid problems. *International Journal for Numerical Methods in Fluids* 1999; **31**:359–392.
30. Codina R, Zienkiewicz OC. CBS versus GLS stabilization of the incompressible Navier–Stokes equations and the role of the time step as stabilization parameter. *Communications in Numerical Methods in Engineering* 2002; **18**:99–112.
31. Codina R. Comparison of some finite element methods for solving the diffusion-convection-reaction equation. *Computer Methods in Applied Mechanics and Engineering* 1998; **156**:185–210.

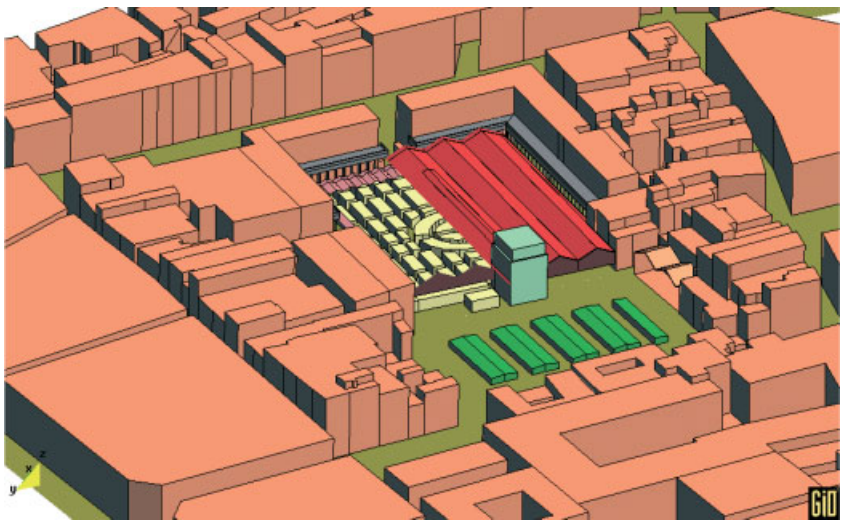

Plate 1. General view of the market problem. Geometry includes the central market together with the surrounding buildings and streets of the neighbourhood. Part of the market roof is printed transparent to appreciate the structure of the internal stalls. The FEM mesh contains approximately 1.5 M elements.

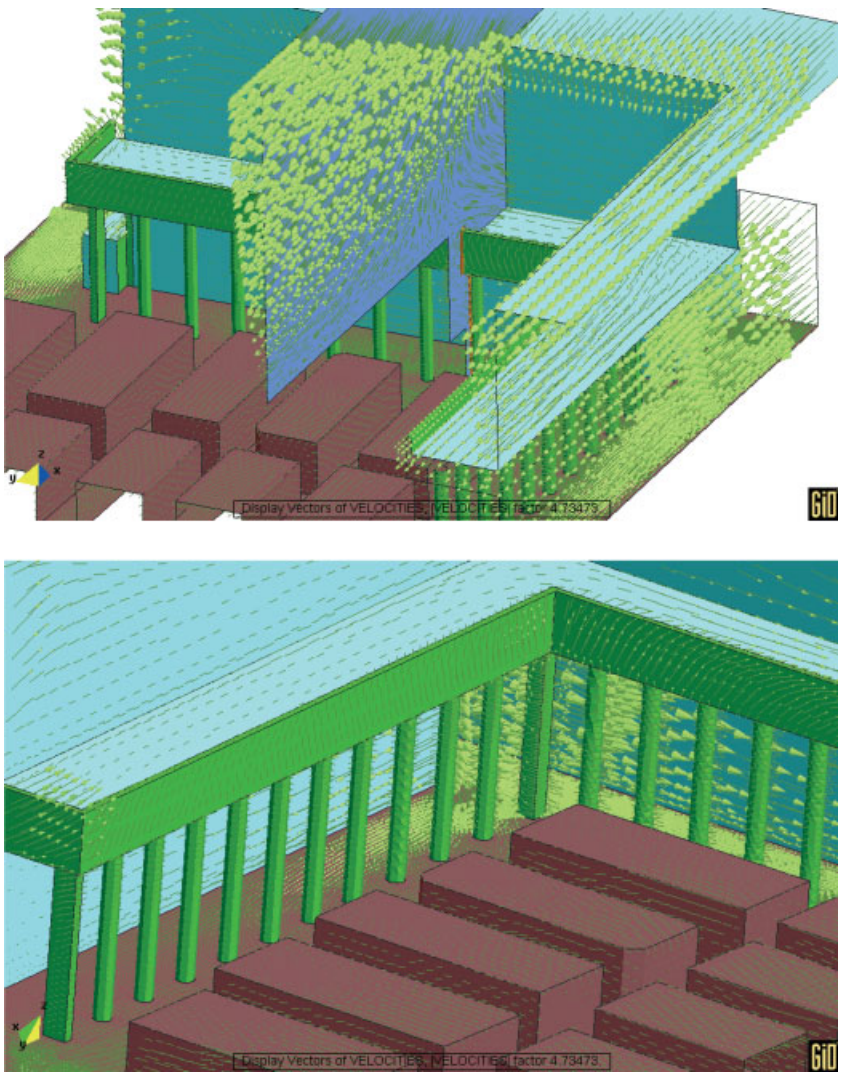

Plate 2. Detail of velocity vectors in a corner of the market.